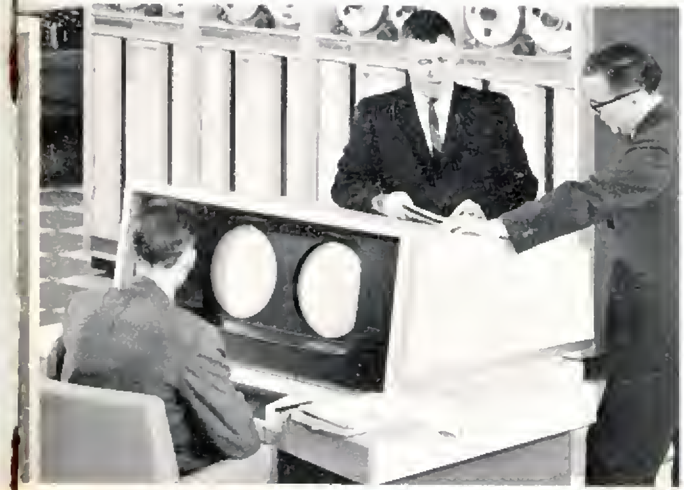
CONTROL DATA
CORPORATION

Documentation Department

3145 Porter Drive
Palo Alto, California 94304

60191900A
APRIL 1968

CONTROL DATA



INSTANT

6400/6500/6600 COMPASS

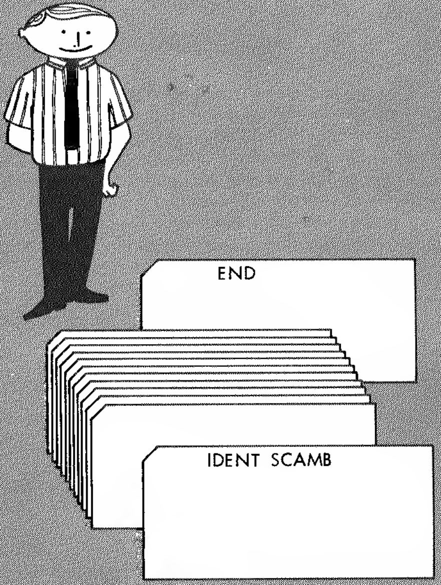
6400/6500/6600 COMPASS

COMPASS is the comprehensive assembly system for the CONTROL DATA® 6400, 6500, and 6600 computers. Running under control of the SCOPE Operating System, COMPASS makes the instruction repertoire of the central and peripheral processors of these machines accessible to programmers who wish to exercise direct control of computer operations. The extensive pseudo instruction repertoire and macro facilities give programmers the advantage of maximum program construction flexibility, including control of the assembly process.

FEATURES

Language Format	Free field
Assembly Mode	Relocatable or absolute; decimal or octal
Conditional Assembly	Character string comparison; expression comparison; symbol attribute tests
Code Duplication	Number of iterations specified by programmer
Data Generation	Data may be declared in subprogram or generated at load time
Macros	System and programmer defined
Micros	Character string definitions
Listing Control	Many output listing options

SOURCE DECK



COMPASS LANGUAGE

Character Set

All characters with display code values from 01-76

Source Statement Fields

LOCATION

The location field must begin in column 1 or 2. An asterisk in column 1 indicates a comment statement; a comma in column 1 indicates a continuation line. Nine continuation cards are permitted.

OPERATION

The operation field begins after the first blank following the location field, not before column 3, and before column 36.

VARIABLE

The variable field begins after the first blank following the operation field and before column 36.

COMMENTS

The comments field begins after the first blank following the variable field, and not before column 36, if the variable field is empty.

Symbols

Up ta 8 characters; the first may not be $ = or numeric and nane may be + - * / blank ar camma.

Linkage symbols are restricted to 7 characters and must begin with an alphabetic character. Linkage symbals must be used for:

Subpragram names
External symbols
Entry points
Camman black names

PP subprogram names may begin with an alphanumeric character and must nat exceed 3 characters.

Names

A name may be formed from 1-8 characters except , → ; or blank, and may not be used in an address expression. Names are used in the following contexts:

Block names
Macro names
Micra names
Instruction bracket names

Absolute Data

Absalute data may be used in data items (LIT and DATA subfields, and literals) or in address expressions as constants.

Register Names

Address registers, 18 bits	A0,A1,...,A7 ar A.n
Index registers, 18 bits	B0,B1,...,B7 ar B.n
Arithmetic and aperand registers, 60 bits	X0,X1,...,X7 or X.n

n is a symbol ar a single digit between 0 and 7

CHARACTER DATA

$$n\begin{bmatrix}C\\H\\A\\R\\L\end{bmatrix}\langle str\rangle \quad ar \quad \begin{bmatrix}C\\H\\A\\R\\L\end{bmatrix}d\langle str\rangle d \quad ar \quad 0\begin{bmatrix}C\\H\\A\\R\\L\end{bmatrix}\langle str\rangle$$

n	Character caunt; if n is preceded by -, character string is complemented; n must not be blank for address canstants
d	Delimiter character; if n is 0 string is terminated by + - * / , Δ for address constants, or by blank ar camma far DATA, LIT, ar a literal
C	Left justify character string with zera fill; twa terminating zeros guaranteed
H	Left justify character string with trailing blanks
A	Right justify character string with leading blanks
R	Right justify character string with leading zeras
L	Left justify character string with trailing zeras
str	Character string excluding d and ; and →

NUMERIC DATA

$\pm\begin{bmatrix}D\\O\\B\end{bmatrix}$ n.n $\begin{bmatrix}E\\EE\end{bmatrix}$ ±n S±n P±n

Omitted signs are assumed positive; modifiers following n.n may appear in any order. Only fixed point values are permitted in peripheral assemblies.

$\begin{bmatrix}D\\O\\B\end{bmatrix}$	Radix; identification of n.n as a decimal or octal number may appear either at beginning or end; if omitted, the radix is determined according to the BASE pseudo instruction. (Assumed decimal if no BASE.)
n.n	Integer and fractional ports; if .n is omitted, the value is integer. The maximum value is 32 significant octal digits or $7.9 \cdot 10^{28}$.
$\begin{bmatrix}E\\EE\end{bmatrix}\pm n$	Single or double precision decimal scale, maximum value 32767
S±n	Binary scale, maximum value 32767
P±n	Binary point position for floating point numbers; the binary point will occur to right of nth bit; the exponent will be adjusted to a value of – (P scale factor).

Default Symbols

=Sname	name is written according to the rules for symbols. name will be defined by COMPASS if not defined by programmer.
=Xname	name will be defined as external symbol if not defined by programmer.

Literals

CHARACTER DATA

Character data literals use the same basic format as data items:

=0 $\begin{bmatrix}C\\H\\A\\R\\L\end{bmatrix}$	<str>Δ	Delimited by subfield end
=n $\begin{bmatrix}C\\H\\A\\R\\L\end{bmatrix}$	<n characters>	Delimited by character count
= $\begin{bmatrix}C\\H\\A\\R\\L\end{bmatrix}$	d<str>d	Delimited by character d

NUMERIC DATA

The numeric value may be integer, fixed point, or floating point data item; if the sign is omitted, the value is assumed positive.

=Bnumeric or =numericB	Octal
=Onumeric or =numericO	Octal
=Dnumeric or =numericD	Decimal

Counters

* or *L	Current value of	Location counter
*O		Origin counter
$		Position counter

Address Expression

Elements ore joined by multiplication or division operotors to form terms; terms ore joined by oddition or subtraction operators to form oddress expressions.

+	Addition
-	Subtraction
*	Multiplication
/	Division

Terms ore evoluated from left to right, then expressions are evoluated from left to right. Two or more contiguous operators are assumed to have intervening elements with a volue of zero. Literals may be used only as the lost term in expressions. Evoluoted expressions must result in on absolute volue, an external value ± constont, or ± relocotoble value ± constant.

ELEMENTS

- Symbols
- Constants
- *, *L, *O, or $
- =Ssymbol or =Xsymbol

TERMS

Within o term the following rules apply:

- Remoinders of division ore dropped
- Division by zero produces a zero result, no error
- Only one relocotable or external element
- Only obsolute values os divisors or to the left of a division
- Omitted last element is assumed zero

CENTRAL PROCESSOR CODES

Xi,Xj,Xk	X register symbols
Ai,Aj,Ak	A register symbols
Bi,Bj,Bk	B register symbols

i,j, and k may have volues 0-7

K	Address expression, 18 bits
n	Absolute address, 6 bits

MNEMONIC		OCTAL	BITS	INSTRUCTION
PS		0000 000000	30	Program stop
RJ	K	0100 K	30	Return jump to K
RE	Bj+K	011j K	30	Read extended core storage
WE	Bj+K	012j K	30	Write extended core storage
XJ	Bj,K	0130 000000	60	Exchange jump
		46000 46000		
JP	K	0200 K	30	Jump to K
JP	Bj+K	020j K	30	Jump to Bj+K
ZR	Xj,K	030j K	30	Jump to K if Xj=0
NZ	Xj,K	031j K	30	Jump to K if Xj≠0
PL	Xj,K	032j K	30	Jump to K if Xj≥0
NG	Xj,K	033j K	30	Jump to K if Xj < 0
IR	Xj,K	034j K	30	Jump to K if Xj is in range
OR	Xj,K	035j K	30	Jump to K if Xj is out or ronge
DF	Xj,K	036j K	30	Jump to K if Xj is definite
ID	Xj,K	037j K	30	Jump to K if Xj is indefinite
ZR	K	0400 K	30	Jump to K
EQ	K	0400 K	30	Jump to K

EQ	Bi,K	04i0 K	30	Jump to K if Bi = 0
ZR	Bi,K	04i0 K	30	Jump to K if Bi = 0
EQ	Bi,Bj,K	04ij K	30	Jump to K if Bi = Bj
NZ	Bi,K	05i0 K	30	Jump to K if Bi ≠ 0
NE	Bi,K	05i0 K	30	Jump to K if Bi ≠ 0
NE	Bi,Bj,K	05ij K	30	Jump to K if Bi ≠ Bj
PL	Bi,K	06i0 K	30	Jump to K if Bi ≥ 0
GE	Bi,K	06i0 K	30	Jump to K if Bi ≥0
GE	Bi,Bj,K	06ij K	30	Jump to K if Bi ≥ Bj
LE	Bj,K	060j K	30	Jump to K if Bj ≤ 0
LE	Bj,Bi,K	06ij K	30	Jump to K if Bj ≤ Bi
NG	Bi,K	07i0 K	30	Jump to K if Bi < 0
LT	Bi,K	07i0 K	30	Jump to K if Bi < 0
LT	Bi,Bj,K	07ij K	30	Jump to K if Bi < Bj
GT	Bj,K	070j K	30	Jump to K if Bi >0
GT	Bj,Bi,K	07ij K	30	Jump to K if Bj > Bi
BXi	Xj	10ijj	15	Xj → Xi
BXi	Xj*Xk	11ijk	15	Log. prod. of Xj and Xk → Xi
BXi	Xj+Xk	12ijk	15	Log. sum of Xj and Xk → Xi
BXi	Xj-Xk	13ijk	15	Log. diff. of Xj and Xk → Xi
BXi	-Xk	14ikk	15	Comp. of Xk → Xi
BXi	-Xk*Xj	15ijk	15	Log. prod. of Xj and Xk comp. → Xi
BXi	-Xk+Xj	16ijk	15	Log. sum of Xk comp. and Xj → Xi
BXi	-Xk-Xj	17ijk	15	Log. diff. of Xk comp. and Xj → Xi
LXi	jk	20ijk	15	Shift Xi left-circular jk places
AXi	jk	21ijk	15	Shift Xi right jk places

LXi	Bj,Xk Xk,Bj	22ijk	15	Shift Xk Bj_{0-5} places → Xi Left shift if Bj positive Right shift if Bj negative
AXi	Bj,Xk Xk,Bj	23ijk	15	Shift Xk Bj_{0-5} places → Xi Right shift if Bj positive Left shift if Bj negative
NXi	Xk	24i0k	15	Normalize Xk → Xi
NXi	Bj,Xk Xk,Bj	24ijk	15	Normalize Xk → Xi Shift count → Bj
ZXi	Xk	25i0k	15	Round and normalize Xk in Xi
ZXi	Bj,Xk Xk,Bj	25ijk	15	Round and normalize Xk → Xi Shift count → Bj
UXi	Xk	26i0k	15	Unpack Xk to Xi
UXi	Bj,Xk Xk,Bj	26ijk	15	Unpack Xk: coefficient → Xi, Exponent → Bj
PXi	Bj,Xk Xk,Bj	27ijk	15	Pack coefficient Xk, Exponent Bj → Xi
FXi	Xj+Xk	30ijk	15	Floating Xj + Xk → Xi
FXi	Xj-Xk	31ijk	15	Floating Xj - Xk → Xi
DXi	Xj+Xk	32ijk	15	Floating double precision Xj + Xk → Xi
DXi	Xj-Xk	33ijk	15	Floating double precision Xj - Xk → Xi
RXi	Xj+Xk	34ijk	15	Rounded floating Xj + Xk → Xi
RXi	Xj-Xk	35ijk	15	Rounded floating Xj - Xk → Xi
IXi	Xj+Xk	36ijk	15	Integer Xj + Xk → Xi
IXi	Xj-Xk	37ijk	15	Integer Xj - Xk → Xi

FXi	Xj*Xk	40ijk	15	Flaating Xj*Xk → Xi
RXi	Xj*Xk	41ijk	15	Raunded flaating Xj*Xk → Xi
DXi	Xj*Xk	42ijk	15	Flaating dauble precisian Xj*Xk → Xi
MXi	n	43in	15	Farm mask af n bits in Xi
FXi	Xj/Xk	44ijk	15	Flooting Xj/Xk → Xi
RXi	Xj/Xk	45ijk	15	Raunded floating Xj/Xk → Xi
NO		4600	15	No aperation
CXi	Xk	47ikk	15	Ones in Xk → Xi
SAi	Aj+K	50ij K	30	Aj + K → Ai
SAi	K	51i0 K	30	K → Ai
SAi	Bj+K	51ij K	30	Bj + K → Ai
SAi	Xj+K	52ij K	30	Xj + K → Ai
SAi	Xj	53ij0	30	Xj → Ai
SAi	[Xj+Bk Bk+Xj]	53ijk	15	Xj + Bk → Ai
SAi	Aj	54ij0	15	Aj → Ai
SAi	[Aj+Bk Bk+Aj]	54ijk	15	Aj + Bk → Ai
SAi	[Aj-Bk -Bk+Aj]	55ijk	15	Aj - Bk → Ai
SAi	Bj	56ij0	15	Bj → Ai
SAi	Bj+Bk	56ijk	15	Bj + Bk → Ai
SAi	-Bk	57i0k	15	-Bk → Ai
SAi	[Bj-Bk -Bk+Bj]	57ijk	15	Bj - Bk → Ai
SBi	Aj+K	60ij K	30	Aj + K → Bi
SBi	K	61i0 K	30	K → Bi
SBi	Bj+K	61ijK	30	Bj + K → Bi
SBi	Xj+K	62ij K	30	Xj + K → Bi

SBi	Xj	63ij0	15	Xj → Bi
SBi	[Xj+Bk Bk+Xj]	63ijk	15	Xj + Bk → Bi
SBi	Aj	64ij0	15	Aj →Bi
SBi	[Aj+Bk Bk+Aj]	64ijk	15	Aj + Bk → Bi
SBi	[Aj-Bk -Bk+Aj]	65ijk	15	Aj - Bk → Bi
SBi	Bj	66ij0	15	Bj → Bi
SBi	Bj+Bk	66ijk	15	Bj + Bk → Bi
SBi	-Bk	67i0k	15	-Bk → Bi
SBi	[Bj-Bk -Bk+Bi]	67ijk	15	Bj - Bk → Bi
SXi	Aj+K	70ij K	30	Aj + K → Xi
SXi	K	71i0K	30	K → Xi
SXi	Bj+K	71ijK	30	Bj + K → Xi
SXi	Xj+K	72ij K	30	Xj + K → Xi
SXi	Xj	73ij0	15	Xj → Xi
SXi	[Xj+Bk Bk+Xj]	73ijk	15	Xj + Bk → Xi
SXi	Aj	74ij0	15	Aj → Xi
SXi	[Aj+Bk Bk+Aj]	74ijk	15	Aj + Bk → Xi
SXi	[Aj-Bk -Bk+Aj]	75ijk	15	Aj - Bk → Xi
SXi	Bj	76ij0	15	Bj → Xi
SXi	Bj+Bk	76ijk	15	Bj + Bk → Xi
SXi	-Bk	76i0k	15	-Bk → Xi
SXi	[Bj-Bk -Bk+Bj]	77ijk	15	Bj - Bk → Xi

PERIPHERAL PROCESSOR CODE

The variable field may cantain index ar oddress volues. Subfields are separated by cammas.

m Address value, 12 bits
c Address value, 18 bits
d Index value, 6 bits
r $-31 \le r \le 31$
M Indexed address (m+d), 18 bits

MNEMONIC		OCTAL	LENGTH	INSTRUCTION
PSN		0000	12	Pass
LJM	m,d	01dd mmmm	24	Lang jump ta M
RJM	m,d	02dd mmmm	24	Return jump ta M
UJN	r	03rr	12	Uncanditianol jump r lacations
ZJN	r	04rr	12	Jump r lacations if (A) = 0
NJN	r	05rr	12	Jump r lacatians if (A) ≠ 0
PJN	r	06rr	12	Jump r locatians if (A) ≥ 0
MJN	r	07rr	12	Jump r locations if (A) < 0
SHN	d	10dd	12	Shift (A) d places, d positive: left circular; d negative: right end aff, na sign extension
LMN	d	11dd	12	Lag. diff. of d and $(A_{5-0}) \rightarrow (A_{6-11})$ unchanged
LPN	d	12dd	12	Log. prad. of d and $(A_{5-0}) \rightarrow A$ (A_{6-11}) are zera.
SCN	d	13dd	12	Clear bits in A_{0-5} far carrespanding anes in d
LDN	d	14dd	12	d → A
LCN	d	15dd	12	-d → A
ADN	d	16dd	12	(A) + d → A
SBN	d	17dd	12	(A) - d → A
LDC	c	20cc cccc	24	c → A
ADC	c	21cc cccc	24	c + (A) → A
LPC	c	22cc cccc	24	Log. prod. af A and C → A
LMC	c	23cc cccc	24	Lag. diff. af A and c → A
EXN	d	260d	12	Exchange jump
MXN	d	261d	12	Monitor exchange jump
RPN		2700	12	Central processar address → A
LDD	d	30dd	12	(location d) → A
ADD	d	31dd	12	(A) + (locatian d) → A
SBD	d	32dd	12	(A) - (locatian d) → A
LMD	d	33dd	12	Log. diff. af A ond lacotian d → A
STD	d	34dd	12	(A_{11-0}) → lacotian d
RAD	d	35dd	12	(A) + (location d) → d and A
AOD	d	36dd	12	(locatian d) + 1 → lacatian d and A
SOD	d	37dd	12	(lacotian d) - 1 → location d and A
LDI	d	40dd	12	((location d)) → A
ADI	d	41dd	12	(A) + ((location d)) → A
SBI	d	42dd	12	(A) - ((location d)) → A
LMI	d	43dd	12	Lag. diff. of A and (lacatian d) → A
STI	d	44dd	12	(A_{11-0}) → ((locotian d))
RAI	d	45dd	12	(A) + ((lacatian d)) → A
AOI	d	46dd	12	((lacation d)) + 1 → (d) and A

SOI	d	47dd	12	((location d)) - 1 → (d) and A
LDM	m,d	50 dd mmmm	24	(M) → A_{0-11}; 0 → A_{12-17}
ADM	m,d	51dd mmmm	24	(A) + (M) → A
SBM	m,d	52dd mmmm	24	(A) - (M) → A
LMM	m,d	53dd mmmm	24	Log. diff. of A and (M) → A
STM	m,d	54dd mmmm	24	(A_{0-11}) → M
RAM	m,d	55dd mmmm	24	(A) + (M) → M and A
AOM	m,d	56dd mmmm	24	(M + 1) → M and A
SOM	m,d	57dd mmmm	24	(M - 1) → M and A
CRD	d	60 dd	12	Read central memory (A) → d,...,d + 4
CRM	m,d	61dd mmmm	24	Read central memory (d) words, beginning with (A) to m,m + 1,...,m + 5d - 1
CWD	d	62dd	12	Write from locations d to d + 4 into central memory address (A)
CWM	m,d	63 dd mmmm	24	Write d words beginning with m to (A) in central memory
AJM	m,d	64dd mmmm	24	Jump to m if channel d active
IJM	m,d	65dd mmmm	24	Jump to m if channel d inactive
FJM	m,d	66dd mmmm	24	Jump to m if channel d full
EJM	m,d	67dd mmmm	24	Jump to m if channel d empty
IAN	d	70dd	12	Input word from channel d to A_{0-11}
IAM	m,d	71dd mmmm	24	Input A words to m from channel d
OAN	d	72dd	12	Output from A on channel d
OAM	m,d	73dd mmmm	24	Output A words from m on channel d
ACN	d	74 dd	12	Activate channel d
DCN	d	75dd	12	Deactivate channel d
FAN	d	76dd	12	External function code (A_{0-11}) → channel d
FNC	m,d	77d mmmm	24	External function code (m) → channel d

PSEUDO INSTRUCTIONS

Assembler Control

IDENT name, origin, entry

Beginning of Subprogram

For an absolute routine, origin may be used to specify the origin of the routine. For o central processor relocatable routine, origin is ignored. For a centrol processor absolute routine, entry specifies the entry point.

END ta

End of Subprogram

A symbol in the location field is assigned the value of last word address +1; ta is an optional transfer address.

ABS

Absolute Assembly

Symbols in the location field or variable field are ignored. Pseudo instructions except ENTRY, EXT, REP and REPI may be used in the subprogram which follows.

PERIPH

PP Assembly

Symbols in the location field or variable field are ignored. Pseudo instructions except ENTRY, EXT, REP and REPI may be used in the subprogram which follows.

BASE $\begin{bmatrix}O\\D\end{bmatrix}$

Numeric Data Mode

O = octal mode; D = decimal mode. Decimal mode is assumed if this pseudo instruction is omitted.

Counter Contro

USE blockname

Block Assignmen

Assemble following instructions into blockname.

Blockname	Type
0 or blank	Nominal subprogram
//	Blank common
*	Block prior to preceding USE
/name/	Labeled common
name	Named local

ORG expression

Origin

Reset origin and location counters. Symbols in expression must be defined previously.

LOC expression

Location Counter

Set location counter to value of address expression.

Linkage Control

ENTRY list of names

Entry Points

Declares entry point names; maximum of seven characters.

EXT list of names

External Names

Declares external names; maximum of seven characters.

Storage Allocation

BSS expression

Storage Reservation

Location field symbol is assigned the value of the location counter, and location and origin counters are incremented by the value of the address field expression.

BSSZ expression

Storage Reservation

Location field symbol is assigned the value of the location counter, and the counters are incremented by the value of the address field expression. At load time, the number of words specified by the expression will be set to zero.

Symbol Definition

symbol EQU expression

Symbol Definition

Symbol in the location field is assigned the value of the address field expression.

symbol SET expression

Symbol Redefinition

Symbol in the location field is redefined to the value of the address field expression.

Data Generation

DATA data items

Data Declaration

Symbol in location field is assigned the value of the location counter. Subfields, separated by commas, may be numeric or character data items.

DIS wc,characters

Display Code Lines

Symbol in the location field is assigned the value of the location counter. wc is the word count: wc · 10 (CP) or wc · 2 (PP) characters beyond the comma are extracted. If wc is blank or zero, the first character after the comma is considered a delimiter, and characters are extracted until the delimiter is again encountered.

LIT data items

Literal Values

Symbol in the location field is assigned the value of the location counter. Up to 100 words of data items, separated by commas, may be included in one LIT instruction

VFD subfields

Field Definition

Symbol in the location field assigns the subfields beginning in a new word. A - in the location field positions the counter at the next quarter word boundary in a CP assembly. Subfields appear as n/v; n is a single element bit count, previously defined and absolute, maximum value of 60; v is an expression. If v is not absolute, the field must be at least 18 bits long, ending at bit number 0, 15, or 30.

REP and REPI subfields

Data Generation

Generate data at load time. Subfields, up to 5, may appear in any order, separated by commas. Subfield format is: specification/non-external address expression

Specifications

S	Source address, mandatory
D	Destination address, S+B if omitted
C	Repetition count, 1 if omitted
B	Code block size, 1 if omitted
I	Increment, B if omitted

Conditional Operations

IFxx field_1, field_2, count

Compare Values

Location field contains instruction bracket name or blank; variable field contains 2 address expressions, separated by commas, for comparison.

Optional count is number of lines to be assembled if comparison is satisfied.

xx	Comparison of Fields
EQ	Equal
NE	Not equal
GT	Greater than
GE	Greater than or equal
LT	Less than
LE	Less than or equal

IFyy count

Test Assembly Environment

Location field contains instruction bracket name or blank. Optional count is number of lines to be assembled if condition is true.

yy	Condition
PP	Peripheral assembly in progress
CP	Central assembly in progress

IF attribute, $\begin{matrix}\text{symbol}\\ \text{expression}\end{matrix}$, count

Test Symbol or Expression Attribute

Location field contains instruction bracket name or blank. Optional count is number of lines to be assembled if attribute is true. A minus before an attribute tests for the negative condition.

Attribute	Test
SET	Symbol defined by SET
ABS	Absolute expression
REL	Common or program relocatable expression
REG	Register name in the expression
COM	Common relocatable expression
EXT	External symbol in expression
LOC	Program relocatable expression
DEF	All symbols in expression defined

IFC xx,dc...cdc...cd,count

Test Character Strings

Location field contains instruction bracket name or blank. Optional count is number of lines to be assembled if comparison is true.

Delimiter is d; c...c is a character string; xx is EQ,NE,GT,GE,LT, or LE

ENDIF

Conditional Assembly Terminator

Location field contains instruction bracket name or blank. ENDIF is ignored if it appears within a line count-controlled range.

List Control

List options

Extent of Listing

Options are separated by commas.

Option	Listing	Nominal Condition
L	List control	on
M	Macro expansion control	off
E	DUP control	off
D	VFD, DATA, DIS; RMT; literals, deferred symbols	off
F	Conditional assembly	off
C	Control cards EJECT, SPACE, TITLE	off
R	Reference table	on
X	XTEXT text	off
S	Systems macro expansion	off
G	Code generation	off
A	Actual assembly list	off
N	Programmer nulls	off
T	SST nulls	off

EJECT

Start New Page

SPACE exp

Skip Line

Skip number of lines indicoted by value of address field expression.

TITLE string

Titling

First title string in subprogram is listed on every page; subsequent TITLEs are subtitles which cause page ejects before listing. The title string begins immediately after the pseudo operation code and continues for 79 columns or to end-of-statement.

Code Duplication

DUP $\text{exp}_1,\text{exp}_2$

Duplication

Location field may contain on instruction brocket name or blonk. Replication count is specified by the value of the first oddress expression. Optional second address specifies number of succeeding lines to be assembled.

ENDD

End Duplicotion

Terminates range if second address field expression was omitted in preceding DUP. Location field may contain an instruction bracket name or blonk.

STOPDUP

Stop Duplicotion

Stops duplicotion at end of current iterotion.

Micros

name MICRO n_1,n_2,dccc...ccd

Micro Definition

The micro string is formed by extracting n_2 characters from ccc...cc, beginning with the character specified by n_1. If n_1 is zero or blank, the character string is empty. If n_2 is zero or blank, the length of the string is delimited by the character d.

≠nome≠

Micro Substitution

Named micro string is substituted by COMPASS wherever ≠nome≠ appears in the line.

Remote Assembly

RMT

Save Code

Instructions up to the next RMT pseudo instruction are saved for later assembly.

HERE

Assemble RMT Code

Saved remote instructions are assembled at this point.

Loader Directives

LCC string

Loader Directives

Character string is possed to binory output file for subsequent recognition by SCOPE loader.

ERR

Forced Error

A fatal error is produced.

file XTEXT record

External Input

Assembles data from named record on named indexed file. If record name is not given, first record of file is used.

SST

System Symbols

Defines system symbols from the system file as if they had been defined by the routine.

Macros

name MACRO arguments

Standard Macro Heading

Arguments must begin with a letter; up to 63 may be listed, separated by special characters: ,.+-*/)($=. Subsequent instructions until ENDM are saved as a macro definition.

MACRO name, arguments

Alternate Macro Heading

The first subfield is the macro name; subsequent subfields are macro arguments. Subfields are separated by commas.

name OPDEF arguments

Special Macro Form Heading

Location field entry is abbreviated description of entire instruction to be recognized as an OPDEF call. Address subfields are formal arguments listed as for MACRO. Provides convenient description of macros in CP machine instruction format.

LOCAL symbols

Local Symbols

Symbols local to macro are separated by commas. Total number of LOCAL symbols and macro arguments must be less than 64.

ENDM

Macro Terminator

Location field contains macro name or blank.

name arguments

Standard Macro call

Non-blank location field forces upper. Arguments are substituted for formal arguments of definition.

name arguments

Alternate Macro Call

Macro call location field is substituted for first parameter in definition argument list.

Assembler Input/Outpu

COMPASS (Input/Output option list)

COMPASS Call

Calls COMPASS Assembler, Input/Output options separated by commas if non-blank.

L	Listing
L or blank	According to internal LIST cantral on file OUTPUT
L=fname	According to internal LIST control on file fname
L=0	Headings only on file OUTPUT

I	Input
I or blank	From file INPUT
I=fname	From file fname

B	Binary Output
B or blank	On file LGO
B=fname	On file fname

S	Systems Text
S or blank	From SYSTEXT
S=rname	From library overlay named rname
S=SCPTEXT	From library overlay SCPTEXT

CHARACTER CODES

Character	Display	External BCD	Hollerith Punch
A	01	61	12-1
B	02	62	12-2
C	03	63	12-3
D	04	64	12-4
E	05	65	12-5
F	06	66	12-6
G	07	67	12-7
H	10	70	12-8
I	11	71	12-9
J	12	41	11-1
K	13	42	11-2
L	14	43	11-3
M	15	44	11-4
N	16	45	11-5
O	17	46	11-6
P	20	47	11-7
Q	21	50	11-8
R	22	51	11-9
S	23	22	0-2
T	24	23	0-3
U	25	24	0-4
V	26	25	0-5
W	27	26	0-6
X	30	27	0-7
Y	31	30	0-8
Z	32	31	0-9
0	33	12	0
1	34	01	1
2	35	02	2
3	36	03	3
4	37	04	4
5	40	05	5
6	41	06	6
7	42	07	7
8	43	10	8
9	44	11	9
+	45	60	12
-	46	40	11
*	47	54	11-8-4
/	50	21	0-1
(	51	34	0-8-4
)	52	74	12-8-4
$	53	53	11-8-3
=	54	13	8-3
blank	55	20	space
,	56	33	0-8-3
.	57	73	12-8-3
≡	60	36	0-8-6
[	61	17	8-7
]	62	32	0-8-2
:	63	00	8-2
≠	64	14	8-4
→	65	35	0-8-5
∨	66	52	11-0
∧	67	37	0-8-7
↑	70	55	11-8-5
↓	71	56	11-8-6
<	72	72	12-0
>	73	57	11-8-7
≤	74	15	8-5
≥	75	75	12-8-5
¬	76	76	12-8-6